TP5 Data Mining: Standard Neural Network and back propagation

Deniz Sungurtekin

June 2020

1 Introduction

Dans ce travail pratique nous allons implémenter un réseaux de neurones artificiel à 2 niveau afin d'effectuer une classification sur un ensemble de donné CIFAR-10 possédant 10 classes. Ce réseau utilisera une fonction d'activation sigmoïde après le premier niveau (hidden layer) puis une "Softmax loss function" pour le niveau de sortie (output layer).

Dans un premier temps nous devons effectuer un "forward pass" avant de calculer l'erreur. Cela consiste uniquement à effectuer de simple opération matricielle permettant de calculer les valeurs de pré-activation de la première et deuxième couche (Z1,Z2) et les valeurs d'activations de la première et deuxième couche (a1,probabilitites), ces valeurs sont indispensable pour calculer l'erreur.

Dans un deuxième temps nous effectuerons le "backward pass" consistant à calculer les dérivés de nos poids (W) et de nos biais (b) afin de les propager dans le réseau. De cette manière, le réseau pourra s'ajuster afin de rendre les poids et les biais optimales.

Finalement, nous effectuerons un "Batch training" où nous mettrons à jour les paramètres W1,b1,W2 et b2 et une prédiction qui consistera uniquement à recalculer notre matrice "probabilitites" avec de meilleur paramètre et d'en extraire les classes possédant les plus grandes probabilités.

2 Résultat et analyse

Pour comprendre l'implémentation de ce réseau, il est donc nécessaire d'introduire la "Softmax function" et la fonction sigmoïde puis leur dérivé.

Dans un premier temps penchons nous sur notre fonction d'activation sigmoïde et sa dérivé:

$$\sigma(x) = \frac{1}{1+exp(-x)}, \sigma'(x) = \frac{exp(-x)}{(1+exp(-x))^2} = exp(-x)*\sigma(x)^2 \quad ou \quad exp(-x) = \frac{1-\sigma(x)}{\sigma(x)}$$

$$On \quad obtient \quad donc: \quad \sigma'(x) = \frac{\sigma(x)^2 * (1-\sigma(x))}{\sigma(x)} = \sigma(x) * (1-\sigma(x))$$

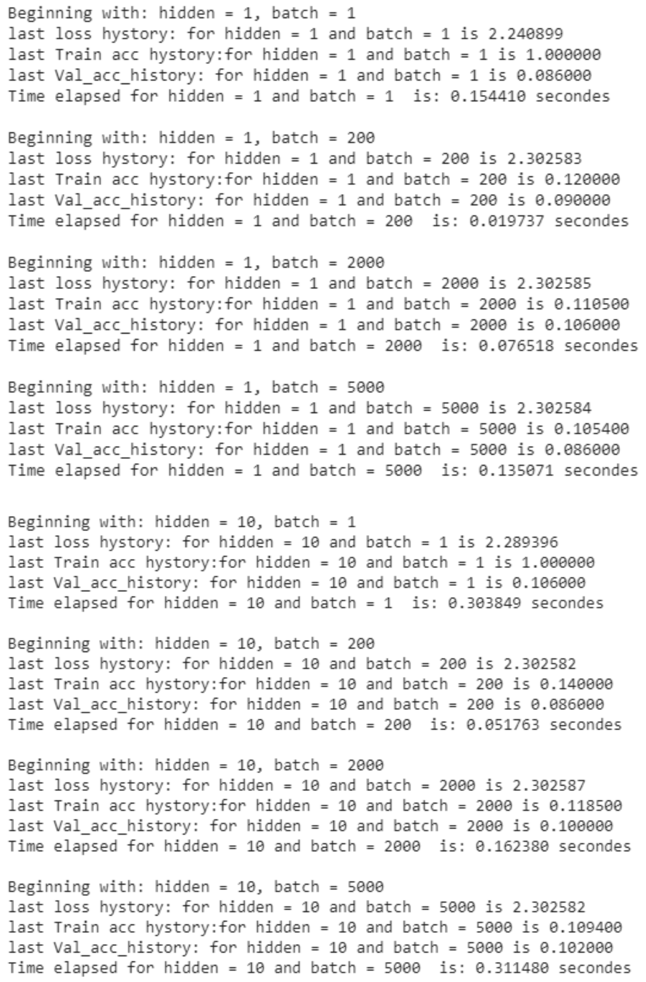
Procédons de la même manière pour la fonction softmax:

$$softmax(x) = \frac{exp(x)}{\sum_{j=1}^{n}(exp(x))}, \frac{\delta softmax(x)}{\delta xi} = \frac{exp(x_i) * (\sum_{j=1}^{n}(exp(x_j)) - exp(x_i) * exp(x_i)}{\sum_{j=1}^{n}(exp(x_j))^2}$$

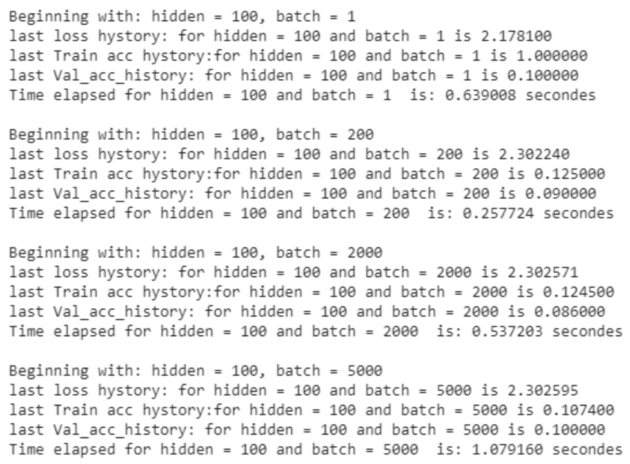
$$= \frac{exp(x_i)}{\sum_{j=1}^{n}(exp(x_j))} - [\frac{exp(x_i)}{\sum_{j=1}^{n}(exp(x_i))}] = softmax(x_i) * [1 - softmax(x_i)]$$

Une fois après avoir remplit les parties manquantes dans le fichier NN.py en utilisant correctement les fonctions softmax et sigmoïde, on peut donc entraîner notre réseaux et se pencher sur l'analyse de nos résultats.

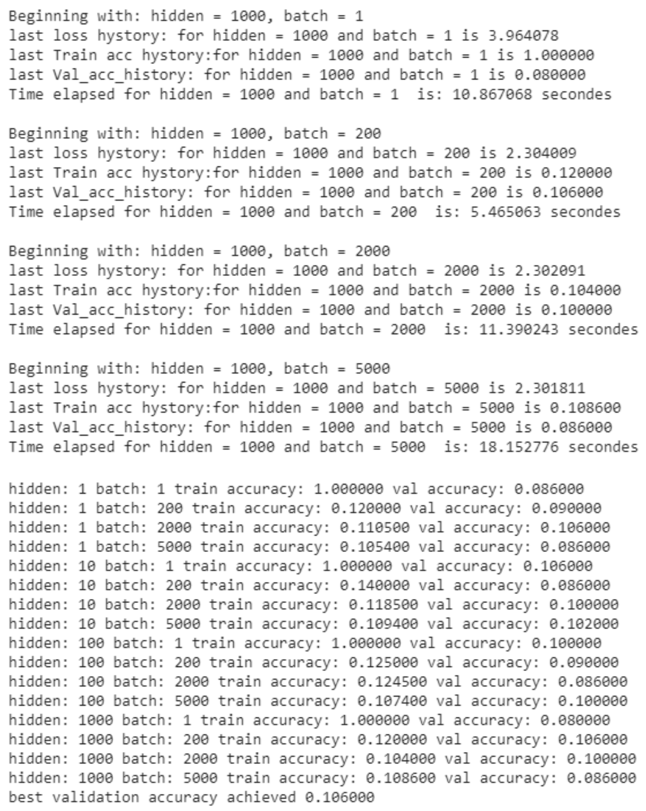
Voici les résultats obtenus avec un learning rate de 0.01 et 500 itérations:



```
Beginning with: hidden = 1, batch = 1
last loss hystory: for hidden = 1 and batch = 1 is 2.240899
last Train acc hystory:for hidden = 1 and batch = 1 is 1.000000
last Val_acc_history: for hidden = 1 and batch = 1 is 0.086000
Time elapsed for hidden = 1 and batch = 1  is: 0.154410 secondes

Beginning with: hidden = 1, batch = 200
last loss hystory: for hidden = 1 and batch = 200 is 2.302583
last Train acc hystory:for hidden = 1 and batch = 200 is 0.120000
last Val_acc_history: for hidden = 1 and batch = 200 is 0.090000
Time elapsed for hidden = 1 and batch = 200  is: 0.019737 secondes

Beginning with: hidden = 1, batch = 2000
last loss hystory: for hidden = 1 and batch = 2000 is 2.302585
last Train acc hystory:for hidden = 1 and batch = 2000 is 0.110500
last Val_acc_history: for hidden = 1 and batch = 2000 is 0.106000
Time elapsed for hidden = 1 and batch = 2000  is: 0.076518 secondes

Beginning with: hidden = 1, batch = 5000
last loss hystory: for hidden = 1 and batch = 5000 is 2.302584
last Train acc hystory:for hidden = 1 and batch = 5000 is 0.105400
last Val_acc_history: for hidden = 1 and batch = 5000 is 0.086000
Time elapsed for hidden = 1 and batch = 5000  is: 0.135071 secondes


Beginning with: hidden = 10, batch = 1
last loss hystory: for hidden = 10 and batch = 1 is 2.289396
last Train acc hystory:for hidden = 10 and batch = 1 is 1.000000
last Val_acc_history: for hidden = 10 and batch = 1 is 0.106000
Time elapsed for hidden = 10 and batch = 1  is: 0.303849 secondes

Beginning with: hidden = 10, batch = 200
last loss hystory: for hidden = 10 and batch = 200 is 2.302582
last Train acc hystory:for hidden = 10 and batch = 200 is 0.140000
last Val_acc_history: for hidden = 10 and batch = 200 is 0.086000
Time elapsed for hidden = 10 and batch = 200  is: 0.051763 secondes

Beginning with: hidden = 10, batch = 2000
last loss hystory: for hidden = 10 and batch = 2000 is 2.302587
last Train acc hystory:for hidden = 10 and batch = 2000 is 0.118500
last Val_acc_history: for hidden = 10 and batch = 2000 is 0.100000
Time elapsed for hidden = 10 and batch = 2000  is: 0.162380 secondes

Beginning with: hidden = 10, batch = 5000
last loss hystory: for hidden = 10 and batch = 5000 is 2.302582
last Train acc hystory:for hidden = 10 and batch = 5000 is 0.109400
last Val_acc_history: for hidden = 10 and batch = 5000 is 0.102000
Time elapsed for hidden = 10 and batch = 5000  is: 0.311480 secondes
```

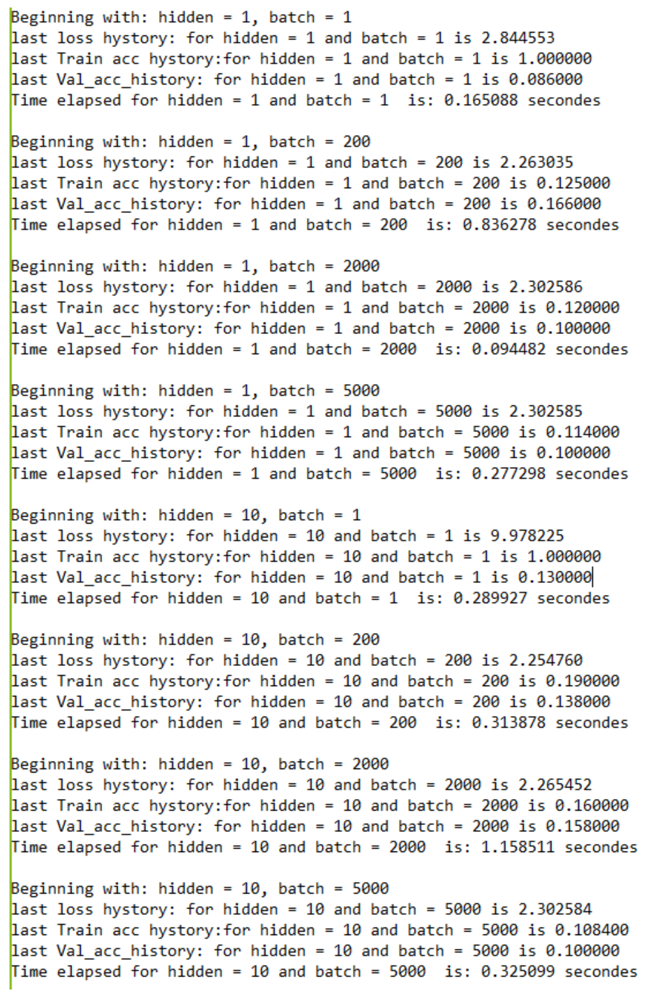


```
Beginning with: hidden = 100, batch = 1
last loss hystory: for hidden = 100 and batch = 1 is 2.178100
last Train acc hystory:for hidden = 100 and batch = 1 is 1.000000
last Val_acc_history: for hidden = 100 and batch = 1 is 0.100000
Time elapsed for hidden = 100 and batch = 1  is: 0.639008 secondes

Beginning with: hidden = 100, batch = 200
last loss hystory: for hidden = 100 and batch = 200 is 2.302240
last Train acc hystory:for hidden = 100 and batch = 200 is 0.125000
last Val_acc_history: for hidden = 100 and batch = 200 is 0.090000
Time elapsed for hidden = 100 and batch = 200  is: 0.257724 secondes

Beginning with: hidden = 100, batch = 2000
last loss hystory: for hidden = 100 and batch = 2000 is 2.302571
last Train acc hystory:for hidden = 100 and batch = 2000 is 0.124500
last Val_acc_history: for hidden = 100 and batch = 2000 is 0.086000
Time elapsed for hidden = 100 and batch = 2000  is: 0.537203 secondes

Beginning with: hidden = 100, batch = 5000
last loss hystory: for hidden = 100 and batch = 5000 is 2.302595
last Train acc hystory:for hidden = 100 and batch = 5000 is 0.107400
last Val_acc_history: for hidden = 100 and batch = 5000 is 0.100000
Time elapsed for hidden = 100 and batch = 5000  is: 1.079160 secondes
```

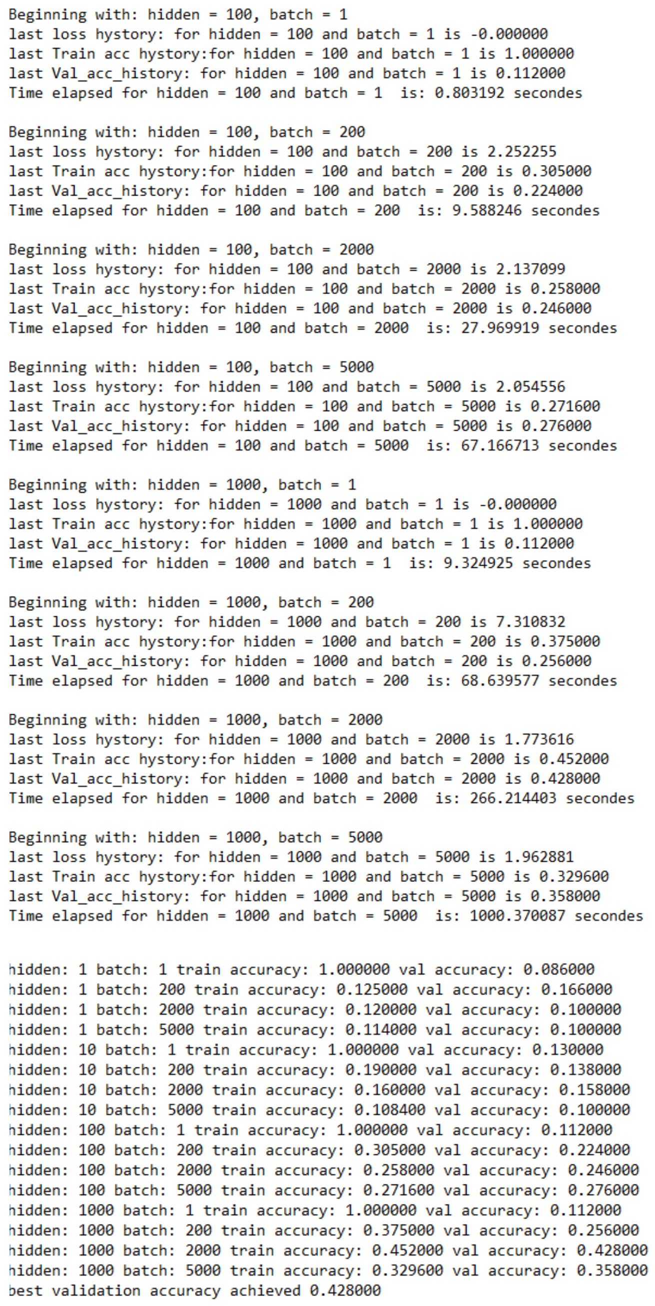


```
Beginning with: hidden = 1000, batch = 1
last loss hystory: for hidden = 1000 and batch = 1 is 3.964078
last Train acc hystory:for hidden = 1000 and batch = 1 is 1.000000
last Val_acc_history: for hidden = 1000 and batch = 1 is 0.080000
Time elapsed for hidden = 1000 and batch = 1  is: 10.867068 secondes

Beginning with: hidden = 1000, batch = 200
last loss hystory: for hidden = 1000 and batch = 200 is 2.304009
last Train acc hystory:for hidden = 1000 and batch = 200 is 0.120000
last Val_acc_history: for hidden = 1000 and batch = 200 is 0.106000
Time elapsed for hidden = 1000 and batch = 200  is: 5.465063 secondes

Beginning with: hidden = 1000, batch = 2000
last loss hystory: for hidden = 1000 and batch = 2000 is 2.302091
last Train acc hystory:for hidden = 1000 and batch = 2000 is 0.104000
last Val_acc_history: for hidden = 1000 and batch = 2000 is 0.100000
Time elapsed for hidden = 1000 and batch = 2000  is: 11.390243 secondes

Beginning with: hidden = 1000, batch = 5000
last loss hystory: for hidden = 1000 and batch = 5000 is 2.301811
last Train acc hystory:for hidden = 1000 and batch = 5000 is 0.108600
last Val_acc_history: for hidden = 1000 and batch = 5000 is 0.086000
Time elapsed for hidden = 1000 and batch = 5000  is: 18.152776 secondes

hidden: 1 batch: 1 train accuracy: 1.000000 val accuracy: 0.086000
hidden: 1 batch: 200 train accuracy: 0.120000 val accuracy: 0.090000
hidden: 1 batch: 2000 train accuracy: 0.110500 val accuracy: 0.106000
hidden: 1 batch: 5000 train accuracy: 0.105400 val accuracy: 0.086000
hidden: 10 batch: 1 train accuracy: 1.000000 val accuracy: 0.106000
hidden: 10 batch: 200 train accuracy: 0.140000 val accuracy: 0.086000
hidden: 10 batch: 2000 train accuracy: 0.118500 val accuracy: 0.100000
hidden: 10 batch: 5000 train accuracy: 0.109400 val accuracy: 0.102000
hidden: 100 batch: 1 train accuracy: 1.000000 val accuracy: 0.100000
hidden: 100 batch: 200 train accuracy: 0.125000 val accuracy: 0.090000
hidden: 100 batch: 2000 train accuracy: 0.124500 val accuracy: 0.086000
hidden: 100 batch: 5000 train accuracy: 0.107400 val accuracy: 0.100000
hidden: 1000 batch: 1 train accuracy: 1.000000 val accuracy: 0.080000
hidden: 1000 batch: 200 train accuracy: 0.120000 val accuracy: 0.106000
hidden: 1000 batch: 2000 train accuracy: 0.104000 val accuracy: 0.100000
hidden: 1000 batch: 5000 train accuracy: 0.108600 val accuracy: 0.086000
best validation accuracy achieved 0.106000
```

On remarque ici des résultats très peu convaincant puisque notre meilleur "validation accuracy" est de 0.106, de plus puisque mon critère d'arrêt survient trop rapidement, il est difficile d'analyser le complexité en temps causé par les paramètres "hidden size" et "batch". Par conséquent j'ai préféré baser mes analyses sur les résultats obtenus avec un learning rate de 1. Voici ces résultats:



```
Beginning with: hidden = 1, batch = 1
last loss hystory: for hidden = 1 and batch = 1 is 2.844553
last Train acc hystory:for hidden = 1 and batch = 1 is 1.000000
last Val_acc_history: for hidden = 1 and batch = 1 is 0.086000
Time elapsed for hidden = 1 and batch = 1  is: 0.165088 secondes

Beginning with: hidden = 1, batch = 200
last loss hystory: for hidden = 1 and batch = 200 is 2.263035
last Train acc hystory:for hidden = 1 and batch = 200 is 0.125000
last Val_acc_history: for hidden = 1 and batch = 200 is 0.166000
Time elapsed for hidden = 1 and batch = 200  is: 0.836278 secondes

Beginning with: hidden = 1, batch = 2000
last loss hystory: for hidden = 1 and batch = 2000 is 2.302586
last Train acc hystory:for hidden = 1 and batch = 2000 is 0.120000
last Val_acc_history: for hidden = 1 and batch = 2000 is 0.100000
Time elapsed for hidden = 1 and batch = 2000  is: 0.094482 secondes

Beginning with: hidden = 1, batch = 5000
last loss hystory: for hidden = 1 and batch = 5000 is 2.302585
last Train acc hystory:for hidden = 1 and batch = 5000 is 0.114000
last Val_acc_history: for hidden = 1 and batch = 5000 is 0.100000
Time elapsed for hidden = 1 and batch = 5000  is: 0.277298 secondes

Beginning with: hidden = 10, batch = 1
last loss hystory: for hidden = 10 and batch = 1 is 9.978225
last Train acc hystory:for hidden = 10 and batch = 1 is 1.000000
last Val_acc_history: for hidden = 10 and batch = 1 is 0.130000
Time elapsed for hidden = 10 and batch = 1  is: 0.289927 secondes

Beginning with: hidden = 10, batch = 200
last loss hystory: for hidden = 10 and batch = 200 is 2.254760
last Train acc hystory:for hidden = 10 and batch = 200 is 0.190000
last Val_acc_history: for hidden = 10 and batch = 200 is 0.138000
Time elapsed for hidden = 10 and batch = 200  is: 0.313878 secondes

Beginning with: hidden = 10, batch = 2000
last loss hystory: for hidden = 10 and batch = 2000 is 2.265452
last Train acc hystory:for hidden = 10 and batch = 2000 is 0.160000
last Val_acc_history: for hidden = 10 and batch = 2000 is 0.158000
Time elapsed for hidden = 10 and batch = 2000  is: 1.158511 secondes

Beginning with: hidden = 10, batch = 5000
last loss hystory: for hidden = 10 and batch = 5000 is 2.302584
last Train acc hystory:for hidden = 10 and batch = 5000 is 0.108400
last Val_acc_history: for hidden = 10 and batch = 5000 is 0.100000
Time elapsed for hidden = 10 and batch = 5000  is: 0.325099 secondes
```

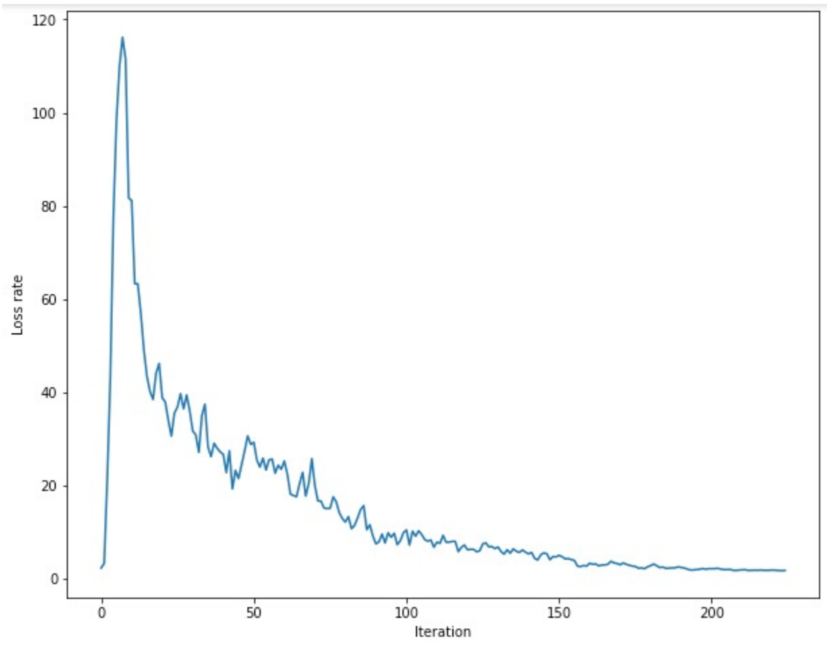


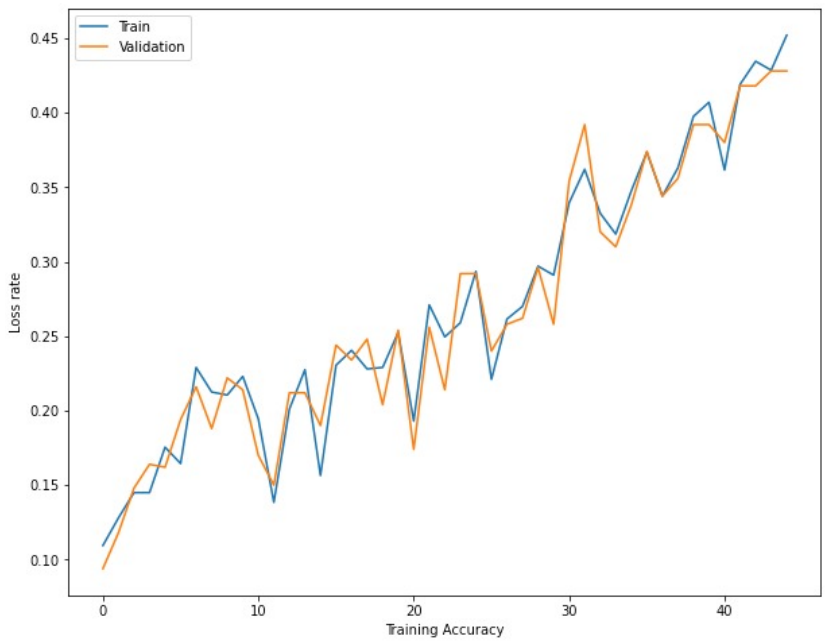
```
Beginning with: hidden = 100, batch = 1
last loss hystory: for hidden = 100 and batch = 1 is -0.000000
last Train acc hystory:for hidden = 100 and batch = 1 is 1.000000
last Val_acc_history: for hidden = 100 and batch = 1 is 0.112000
Time elapsed for hidden = 100 and batch = 1  is: 0.803192 secondes

Beginning with: hidden = 100, batch = 200
last loss hystory: for hidden = 100 and batch = 200 is 2.252255
last Train acc hystory:for hidden = 100 and batch = 200 is 0.305000
last Val_acc_history: for hidden = 100 and batch = 200 is 0.224000
Time elapsed for hidden = 100 and batch = 200  is: 9.588246 secondes

Beginning with: hidden = 100, batch = 2000
last loss hystory: for hidden = 100 and batch = 2000 is 2.137099
last Train acc hystory:for hidden = 100 and batch = 2000 is 0.258000
last Val_acc_history: for hidden = 100 and batch = 2000 is 0.246000
Time elapsed for hidden = 100 and batch = 2000  is: 27.969919 secondes

Beginning with: hidden = 100, batch = 5000
last loss hystory: for hidden = 100 and batch = 5000 is 2.054556
last Train acc hystory:for hidden = 100 and batch = 5000 is 0.271600
last Val_acc_history: for hidden = 100 and batch = 5000 is 0.276000
Time elapsed for hidden = 100 and batch = 5000  is: 67.166713 secondes

Beginning with: hidden = 1000, batch = 1
last loss hystory: for hidden = 1000 and batch = 1 is -0.000000
last Train acc hystory:for hidden = 1000 and batch = 1 is 1.000000
last Val_acc_history: for hidden = 1000 and batch = 1 is 0.112000
Time elapsed for hidden = 1000 and batch = 1  is: 9.324925 secondes

Beginning with: hidden = 1000, batch = 200
last loss hystory: for hidden = 1000 and batch = 200 is 7.310832
last Train acc hystory:for hidden = 1000 and batch = 200 is 0.375000
last Val_acc_history: for hidden = 1000 and batch = 200 is 0.256000
Time elapsed for hidden = 1000 and batch = 200  is: 68.639577 secondes

Beginning with: hidden = 1000, batch = 2000
last loss hystory: for hidden = 1000 and batch = 2000 is 1.773616
last Train acc hystory:for hidden = 1000 and batch = 2000 is 0.452000
last Val_acc_history: for hidden = 1000 and batch = 2000 is 0.428000
Time elapsed for hidden = 1000 and batch = 2000  is: 266.214403 secondes

Beginning with: hidden = 1000, batch = 5000
last loss hystory: for hidden = 1000 and batch = 5000 is 1.962881
last Train acc hystory:for hidden = 1000 and batch = 5000 is 0.329600
last Val_acc_history: for hidden = 1000 and batch = 5000 is 0.358000
Time elapsed for hidden = 1000 and batch = 5000  is: 1000.370087 secondes


hidden: 1 batch: 1 train accuracy: 1.000000 val accuracy: 0.086000
hidden: 1 batch: 200 train accuracy: 0.125000 val accuracy: 0.166000
hidden: 1 batch: 2000 train accuracy: 0.120000 val accuracy: 0.100000
hidden: 1 batch: 5000 train accuracy: 0.114000 val accuracy: 0.100000
hidden: 10 batch: 1 train accuracy: 1.000000 val accuracy: 0.130000
hidden: 10 batch: 200 train accuracy: 0.190000 val accuracy: 0.138000
hidden: 10 batch: 2000 train accuracy: 0.160000 val accuracy: 0.158000
hidden: 10 batch: 5000 train accuracy: 0.108400 val accuracy: 0.100000
hidden: 100 batch: 1 train accuracy: 1.000000 val accuracy: 0.112000
hidden: 100 batch: 200 train accuracy: 0.305000 val accuracy: 0.224000
hidden: 100 batch: 2000 train accuracy: 0.258000 val accuracy: 0.246000
hidden: 100 batch: 5000 train accuracy: 0.271600 val accuracy: 0.276000
hidden: 1000 batch: 1 train accuracy: 1.000000 val accuracy: 0.112000
hidden: 1000 batch: 200 train accuracy: 0.375000 val accuracy: 0.256000
hidden: 1000 batch: 2000 train accuracy: 0.452000 val accuracy: 0.428000
hidden: 1000 batch: 5000 train accuracy: 0.329600 val accuracy: 0.358000
best validation accuracy achieved 0.428000
```

On remarque déjà une nette amélioration avec un meilleur "validation accuracy" de 0.428.

En observant ces résultats, on déduit que la "validation accuracy" n'est pas forcément la meilleur avec les plus grandes valeurs de "hidden size" et de "batch size” puisque le cas avec hidden size = 1000 et batch size = 5000 présentent de moins bon résultat qu’avec hidden size = 1000 et batch size = 2000. Il existe donc un moment où l’on surentraîne notre réseaux nuisant à nos résultats. De plus, en testant mon code sans critère d’arrêt, j’ai pu observer l’impact des paramètres ”hidden size” et ”batch size” sur le temps de calcul engendré et j’ai remarqué une augmentation linéaire de la complexité en temps du calcul.

Voici donc mes graphiques du meilleur model:





On remarque pour notre erreur qu'elle baisse très rapidement sur les 50 premières itérations, puis beaucoup plus lentement ensuite jusqu'à une certaine stabilisation.

En ce qui concerne la training et validation accuracy, on remarque qu'elle s'améliore quant à elle de manière linéaire en possédant presque toujours des valeurs très similaire.

3 Conclusion

Dans ce travail, on a pu donc comprendre avec un exemple simple comment se construit un réseau, en particulier le forward pass et le backward pass. C'est surtout dans le backward pass que réside la complexité de cette méthodologie qui ajuste de manière intelligente ses paramètres afin de mieux prédire les labels en question. De plus, l'utilisation de la fonction softmax nous permet de distinguer plus de deux classes contrairement aux anciennes méthodes vu dans le cours. Cependant, ce travail pratique ayant pour but notre compréhension du fonctionnement des réseaux de neurones, il est je pense normale d'obtenir des résultats aussi peu exacte puisque nous possédons uniquement un seul hidden layer.